
常见编译错误

引言

刚开始接触代码的时候的由于自己对知识点的不了解，达不到准确的应用，所以会发生一些常见的编译异常。这是很多新手都会遇到的问题，所以小慕为大家准备了一些常见的编译错误和解决方法。

Tips:如果在编程中出现编译错误，可以将光标移动到代码飘红处，然后会出现如下提示，建议同学仔细阅读提示中的英文(如果有不明白的地方，可以去使用翻译软件进行翻译)

1、重复定义相同变量名（导致编译错误）

示例代码

```
double d1=123;
System.out.println("d1="+d1);
//变量间的赋值
double d1=d;
System.out.println("d2="+d2);
```

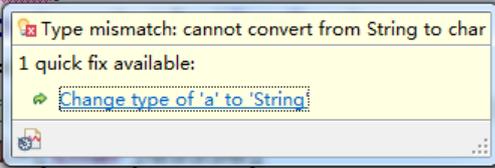
错误原因：该错误是 d1 这个变量重复定义了，也就是说一山不容二虎，如果同为方法内的变量，就不可以重复定义相同的变量名。改为 double d2 = d 即可。



2、类型不匹配:无法将字符串转换为字符

示例代码

```
public class CharDemo {  
  
    public static void main(String[] args) {  
        // 定义一个变量存放字符 'a'  
        char a="a";  
        System.out.println("a="+a);  
        char ch="1";  
        //如果字面量是字符串，需要进行强制类型转换。  
        char ch1="1";  
        System.out.println("ch="+ch);  
        //定义变量存放unicode编码表示的字符  
        char c='\u005d';  
        System.out.println("c="+c);  
    }  
}
```



错误原因

char 类型的数据表示一个字符，它可以是 a，或者是 1，但是不能用双引号老表示，只能用单引号来表示。双引号表示的是字符串，应该用 String 类型的数据来接收。

3、final 关键字修饰的常量无法再次赋值

示例代码

```
public static void main(String[] args) {
    int m=5;
    final int N=6;
    m=10;
    final double PI=3.14159;
    final double MIN_VALUE=0;
    N = 4;
}
```

The final local variable N cannot be assigned. It must be blank and not using a compound assignment

1 quick fix available:

Remove 'final' modifier of 'N'

错误原因

final 关键字修饰的常量是无法再进行赋值的，否则会出现上图中的编译错误。

4、超出类型取值范围的错误

示例代码

```
float f1=103948583923948;
```

```
System.out.println(f1);
```

The literal 103948583923948 of type int is out of range

错误原因

int 类型数据的取值范围是-2的31次方到2的31次方-1 所以 103948583923948

超出了 int 类型的取值范围，应改用 long 类型（长整型），需要在 103948583923948

后加一个 L。